Excel for Exchange of Indivisible Goods: A Users' Manual for Ver.3*

*This material was written by Shuya Abe (Graduate School of Information Science, Osaka University) and Naoki Watanabe (Graduate School of Business Administration, Keio University), and it is published by Keio Business School. Inquiries about reproducing the case should be referred to Keio Business School (4-1-1 Hiyoshi Kohoku, Yokohama, Kanagawa 223-8526; Phone: +81-45-564-2444; E-Mail: case@kbs.keio.ac.jp) To order copies of the case, go to the website (http://www.kbs.keio.ac.jp). Copyright©Shuya Abe and Naoki Watanabe (1st ed., October 2018)

Contents

1 Introduction

This material is a users' manual for a software *Excel for Exchange of Indivisible Goods* which runs on a computer. This section outlines the operating environment of the software and exchange of indivisible goods, providing some notes on the software.

1.1 Recommended Operating Environment

Excel for Exchange of Indivisible Goods version 3 (exchange.xlsm) was developed by the authors and is currently (January 2025) available from a github repository. The URL is as follows.

`https://github.com/shuya-abe/excel-for-exchange`

In the repository, click on the green button "Code" to open a drop-down list. Then, choose "Download Zip." This is the simplest way to download the file.

This program uses the macro function of Excel, and thus the download may be blocked by companies, research and educational institutions in some cases. If so, please download it in users' own personal Internet environments.

The screen-shots in this manual were taken in the environment shown in Table 1.

Table 1: Environment of screen-shots

OS	Windows 10 Enterprise
Excel	Microsoft Office 2016 Excel
CPU	Intel(R) Core(TM) i7-3770K CPU @ 3.50GHz 3.90GHz
Memory	8 GB
Disk	1 TB

Excel for Exchange of Indivisible Goods runs on computers that can use Excel VBA (Visual Basic for Applications). The recommended environment is listed in Table 2.

1.2 Exchange of Indivisible Goods

Consider, for example, an assignment of rooms in residence halls to college students. These rooms are different in floor plan, sunshine hours, location in the residence halls, age of the buildings, appliances and equipments, rent,

Table 2: Recommended environment

OS	Windows 7–CMac OS X 10.8–
Excel	Microsoft Office 2013–CMicrosoft Office for Mac 2016
CPU	no requirement if Excel works
Memory	at least 2GB
Disk	no in particular

and so on. Students have different preferences over the rooms. Suppose that a single room which may not be shared with other students (i.e., each room is indivisible) is initially allocated by their college to each student, but that students are allowed to freely exchange their rooms after the initial allocation. Any side payments among students are prohibited here. Then, what is a desirable reassignment in this situation? This is called a house-swapping problem.

In Table 3, for example, the preferences of six students over their rooms are listed, where the upper bars on lower-case letters of the Roman alphabet mark their currently assigned room; C strictly prefers room e most and likes his or her own house c more than other rooms a, d, and b.

Table 3: Table of Preferences over rooms

A:	c	e	$\bar{a}$	b	d
B:	$\bar{b}$	a	c	e	d
C:	e	$\bar{c}$	a	d	b
D:	c	a	b	e	$\bar{d}$
E:	d	c	b	$\bar{e}$	a

Treating house owners (or college students) as traders of the indivisible houses (or rooms in residence halls), Shapley and Scarf (1974) formulated this problem as exchange of indivisible goods and introduce an algorithm to compute a desirable reassignment in the problem.[1] In every house-swapping problem, there is always only one reassignment of the houses so that no group could have improved even one member's position without making some other member worse off. This unique reassignment is called **strict core allocation** in game theory, given a list of traders' strict preferences (rankings

[1]The theory of exchange of indivisible goods is currently applied to various real practices in many countries: the reassignment of rooms in residence halls to college students, the matching of kidney donors and patients, etc.

representing their preferences without any ties) over houses.

Excel for Exchange of Indivisible Goods computes the strict core allocation by using the **TTC algorithm** developed by David Gale and introduced in Shapley and Scarf (1974). The computation process is not described here but explained in the Appendix.

1.3 Some Notes

(1) *Excel for Exchange of Indivisible Goods* consists of 3 Excel sheets (man, result, analyze) and 1 Visual Basic for Applications (VBA) program. If assistance is needed for reading and writing VBA codes, refer to an appropriate textbook.

(2) In the inputs and outputs of this software, house owners and their houses are represented by numerical IDs, respectively in inputs and outputs of this software.

(3) Users need to input numerical values in **Arabic numerals (not double-byte characters) without multiple numbers and without missing values** for initial settings, traders' preferences over houses.

(4) *Excel for Exchange of Indivisible Goods* uses a program written in Visual Basic. Thus, unlike editing a regular Excel sheet, users cannot restore states by pressing the "backspace" key or pressing the "Z" key while holding down the "Ctrl" key. It is possible to return the Excel file to the intended state before editing by closing the Excel file without saving the current state.

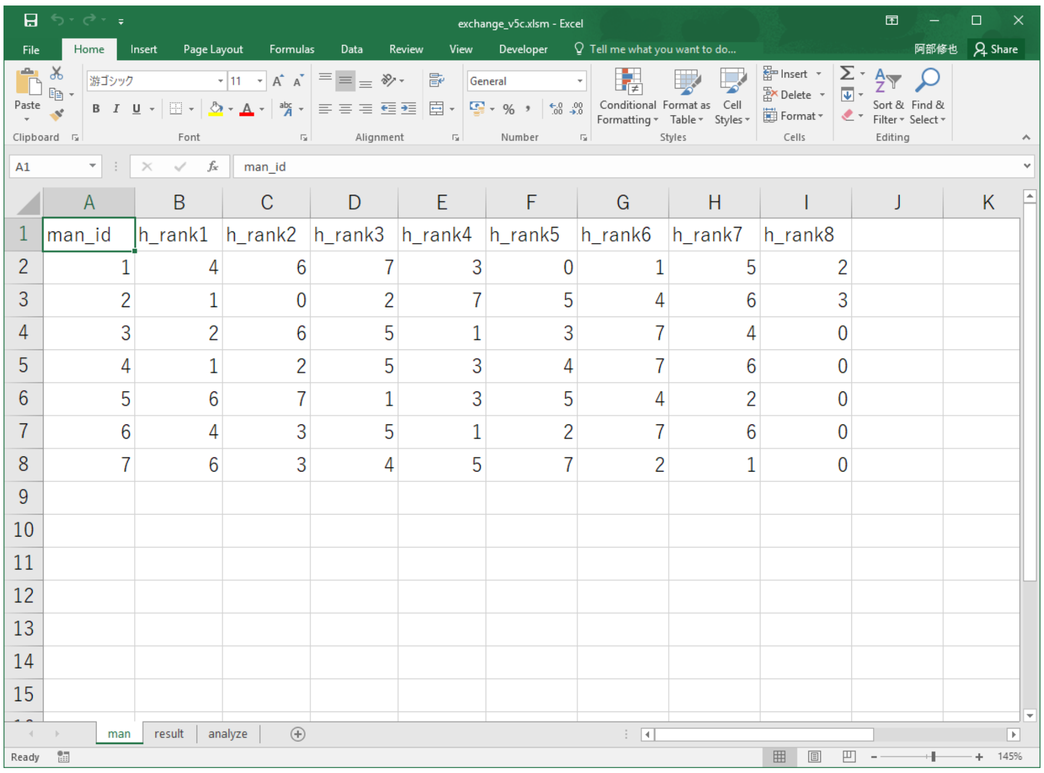
(5) The computation results of this software are overwritten each time the program is run. Thus, users need to make a copy of the computation results in another file.

2 Configuration

Every house owner has his or her strict preference over all houses, i.e., ranking of houses without ties. In this software, houses are denoted by house IDs in Arabic numerals, and house owners are denoted by house owner IDs also in Arabic numerals.

2.1 man sheet

Input house owners' preferences over houses represented by house IDs in this sheet. See Fig.1, where 7 house owners all report their preferences over 7 houses and a null house. All houses which are not preferred to the null house by a house owner are not assigned to the house owner; when a house owner is reassigned to a house which is not preferred to the null house, he or she will choose free disposal of the house. The null house is denoted by 0 as its house ID.



man_id	h_rank1	h_rank2	h_rank3	h_rank4	h_rank5	h_rank6	h_rank7	h_rank8
1	4	6	7	3	0	1	5	2
2	1	0	2	7	5	4	6	3
3	2	6	5	1	3	7	4	0
4	1	2	5	3	4	7	6	0
5	6	7	1	3	5	4	2	0
6	4	3	5	1	2	7	6	0
7	6	3	4	5	7	2	1	0

Figure 1: man sheet

Rows in column A (man_id): Input house owner (man) IDs in ascending order **without multiple numbers and without missing values**.

Rows in column B and thereafter (h_rank): Each row is used for a house owner. Input the house ID man i prefers at the jth rank in the cell corresponding to row (man_id) i and column (h_rank) j. In Fig.1, e.g., the second-best house (h_rank2) for a house owner with ID=3 (man_id=3) is a house with ID=6. The value put in cell 4C is thus 6.

randomize button: Use this button to generate the house owners' preferences randomly. When the preferences are randomly generated, the null house is put at the last rank for every house owners. In Fig.1, the list of preferences shown there was input by use's own hand and the randomize button is outside the frame of the screen shot. There is another randomize button in Subsect.2.2, which functions in the same way.

By using TTC algorithm, *Excel for Exchange of Indivisible Goods* computes a unique reassignment of houses called the strict core allocation in game theory, given a list of house owners' strict preferences over houses. In the strict core allocation, no house owner is reassigned to a house which is at the lower rank than his or her own house. This property is called individual rationality. Thus, there is no need for house owners to report their rankings of those houses which are not preferred to their own houses. Even if the house IDs for those house are missing in the man sheet, this software computes the strict core allocation.

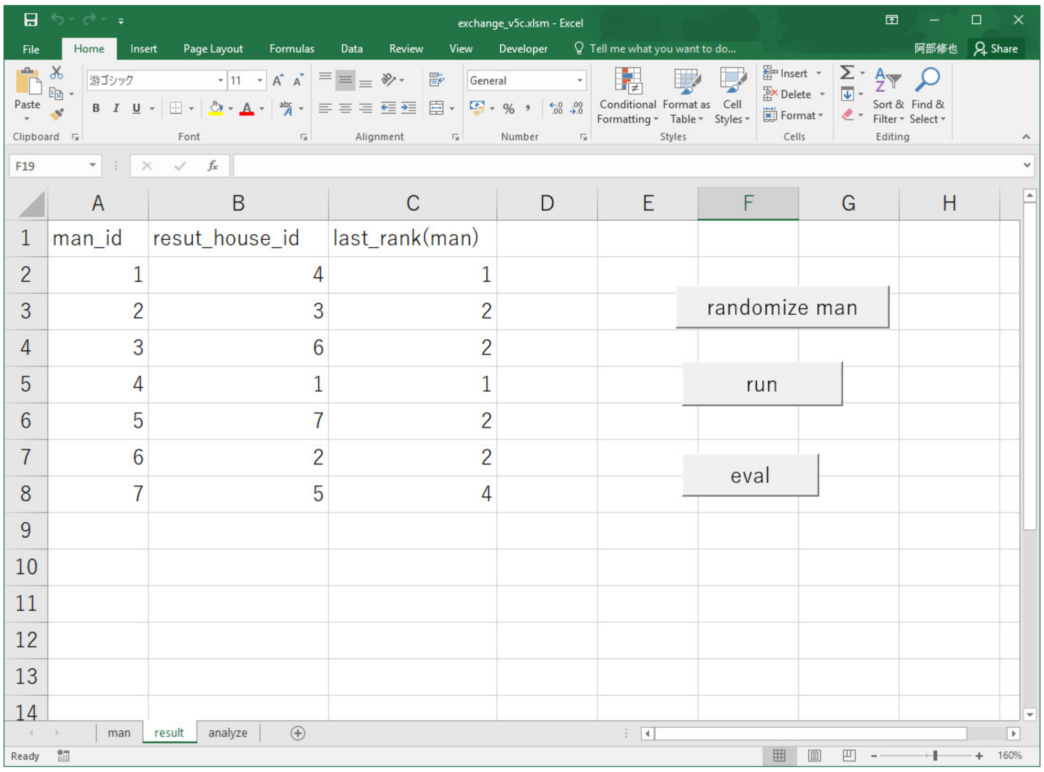
When you use a function that examines whether a reassignment given in your own hand is (Pareto) efficient, however, make a complete list of rankings in the man sheet, because the function needs such information. If there are missing values in preferences in the man sheet, *Excel for Exchange of Indivisible Goods* will display a run-time error message, when you press the eval button in the result sheet, which is described in the next subsection. Note that the strict core allocation is always (Pareto) efficient.

2.2 result sheet

The computation results are shown in the result sheet. The buttons in this sheet are used to run each computation. See Fig.2, which shows a reassignment of 7 houses among 7 house owners and rankings of houses assigned to them according to their preferences. As for TTC algorithm, see the Appendix for the illustrations with an example.

Rows in column A (man_id): House owner IDs are shown in ascending order.

Rows in column B (result_house_id): Houses reassigned to house owners are shown with house IDs, respectively. In Fig.2, e.g., a house owner with ID=3 (man_id=3) is reassigned to a house with ID=6 (result_house_id=6).



man_id	resut_house_id	last_rank(man)
1	4	1
2	3	2
3	6	2
4	1	1
5	7	2
6	2	2
7	5	4

Figure 2: result sheet

Rows in column C (last_rank (man)): The rank of the house that is reassigned to a house owner in his or her preference is shown for each house owner. In Fig.2, e.g., a house owner with ID=3 is reassigned to a house with ID=6, which is the 2nd rank of his or her preference (last_rank (man)=2). The values put in cells 4B and 4C are thus 6 and 2, respectively.

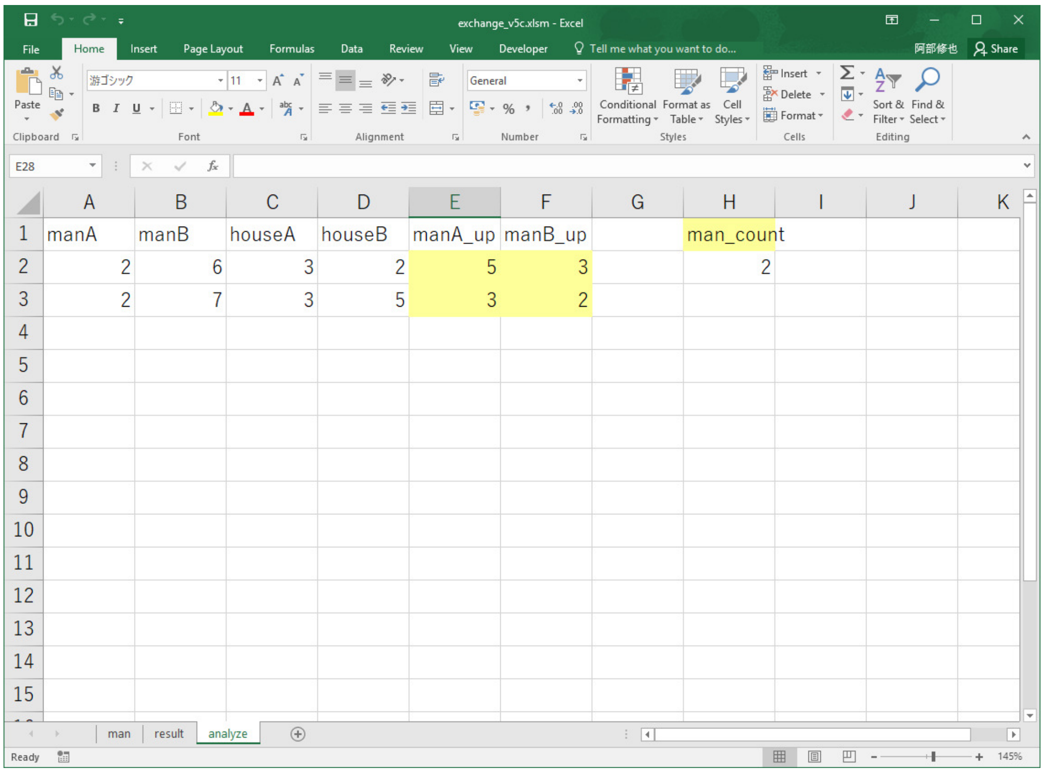
randomize man **button:** The same function button that appears in the man sheet (Subsect.2.1). Use this button to generate the house owners' preferences randomly. When the preferences are randomly generated, the null house is put at the last rank for every house owners. The list of preferences shown in Fig.1 was input by use's own hand.

run **button:** Press to compute the unique strict core allocation by using the TTC algorithm.

eval **button:** Press when the (Pareto) efficiency of a reassignment shown in the result sheet or an allocation input by your own hand there is examined according to house owners' preferences input in the man sheet (Subsect.2.1).

2.3 analyze sheet

Consider a situation where a pair of house owners exchange their houses, given an initial allocation or a reassignment (an allocation). If both house owners are better off by that exchange, then the allocation is not (Pareto) efficient. When you press eval button in the result sheet, *Excel for Exchange of Indivisible Goods* starts searching for those pairs examining every pair of house owners. (It is not necessarily true that an allocation is Pareto efficient, when there is no pair of house owners who are both better off by that exchange.) The examination result is shown in the analyze sheet.



	A	B	C	D	E	F	G	H
1	manA	manB	houseA	houseB	manA_up	manB_up		man_count
2	2	6	3	2	5	3		2
3	2	7	3	5	3	2		

Figure 3: analyze sheet

Fig.3 lists all pairs of house owners who are both better off by exchange of their initially allocated or reassigned houses. The numbers colored in yellow indicate the increase in their ranking over houses by that exchange, which implies that the initial allocation or a reassignment is not (Pareto) efficient.

The number of those observations is counted and shown in the upper-right part of the sheet (man_count).

Rows in column A (manA) and column B (manB): All pairs of house owners who can be better off by exchange of their houses are shown with their IDs.

Rows in column C (houseA) and column D (houseB): Houses initially allocated or reassigned to the house owners listed in man A and man B are shown with their IDs.

Rows in column E (manA_up) and column F (manB_up): The increments of ranks at the time when manA obtains houseB and manB obtains house A, respectively.

Column H (man_count): The number of those observations is shown in the upper-right part of the sheet.

In Fig.3, e.g., a house with ID=3 is allocated to a house owner with ID=2 and another house with ID=5 is allocated to another house owner with ID=7. When they exchange their houses initially allocated or reassigned, a house owner with ID=2 obtains a house which is 3-rank higher than the one he or she has, and a house owner with ID=7 obtains a house which is 2-rank higher than he or she has. There are 2 observations of such a pairs, which is shown in man_count.

3 Using *Excel for Exchange of Indivisible Goods*

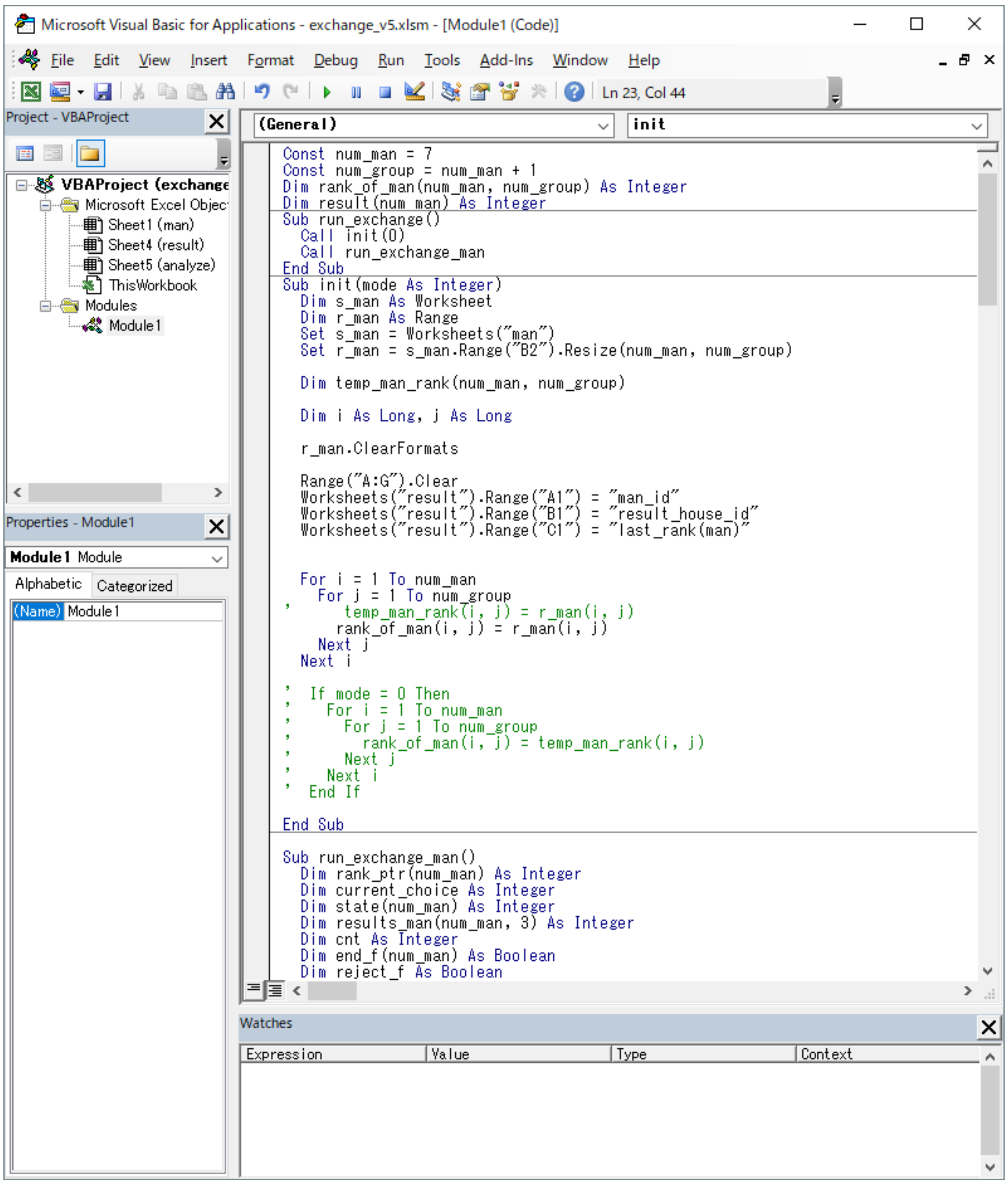
3.1 Getting Started

Initially, set the number of house owners, not in Excel sheets, but in the VBA window (Fig.4). In the default, the number of house owners is set as 7.

1. Click on the **Visual Basic** button on the **Developer** ribbon.

2. Find **VBAProject (exchange_v3 (version 1).xlsb)** by **Project** explorer.

3. Open **Module 1** in the **Standard Module** that is under **VBAProject (exchange_v3 (version 1).xlsb)**.



```
Const num_man = 7
Const num_group = num_man + 1
Dim rank_of_man(num_man, num_group) As Integer
Dim result(num_man) As Integer
Sub run_exchange()
  Call init(0)
  Call run_exchange_man
End Sub
Sub init(mode As Integer)
  Dim s_man As Worksheet
  Dim r_man As Range
  Set s_man = Worksheets("man")
  Set r_man = s_man.Range("B2").Resize(num_man, num_group)

  Dim temp_man_rank(num_man, num_group)

  Dim i As Long, j As Long

  r_man.ClearFormats

  Range("A:G").Clear
  Worksheets("result").Range("A1") = "man_id"
  Worksheets("result").Range("B1") = "result_house_id"
  Worksheets("result").Range("C1") = "last_rank(man)"


  For i = 1 To num_man
    For j = 1 To num_group
'       temp_man_rank(i, j) = r_man(i, j)
      rank_of_man(i, j) = r_man(i, j)
    Next j
  Next i

'  If mode = 0 Then
'    For i = 1 To num_man
'      For j = 1 To num_group
'        rank_of_man(i, j) = temp_man_rank(i, j)
'      Next j
'    Next i
'  End If

End Sub

Sub run_exchange_man()
  Dim rank_ptr(num_man) As Integer
  Dim current_choice As Integer
  Dim state(num_man) As Integer
  Dim results_man(num_man, 3) As Integer
  Dim cnt As Integer
  Dim end_f(num_man) As Boolean
  Dim reject_f As Boolean
```

Figure 4: VBA window

4. Change the numbers in Const num_man to set the number of house owners. This is the first line in **Module 1**.

5. Save to the same file and close the VBA window.

3.2 Running the Program

1. Complete the initial setting for the numbers of workers and divisions in the VBA window.
2. Input numerical data on house owners' preferences in the (1) man sheet. Click on randomize button if you generate the house owners' preferences randomly
3. Open the (2) result sheet.
4. Click on the run button for computing a matching with the TTC algorithm.
5. View computation results in the result sheet.
6. Click on the eval button for examining the efficiency of an reassignment shown in the result sheet. Input numerical data on an allocation by your own hand. Then, you can examine the efficiency of the allocation you determined by yourself.
7. Open the (3) analyze sheet and view the examination result in the analyze sheet.

3.3 A Tips

This software has a function to examine the inefficiency of the reassignment shown in the result sheet (Subsect.2.2) based on house owners' preferences input in the man sheet (Subsect.2.1). You can enter the data of actual allocation you have in the result sheet and press eval button, without pressing run. The inefficiency of the actual allocation can be examined in the way described above, given house owners' preferences over houses. The strict core allocation calculated by the TTC algorithm is (Pareto) efficient.

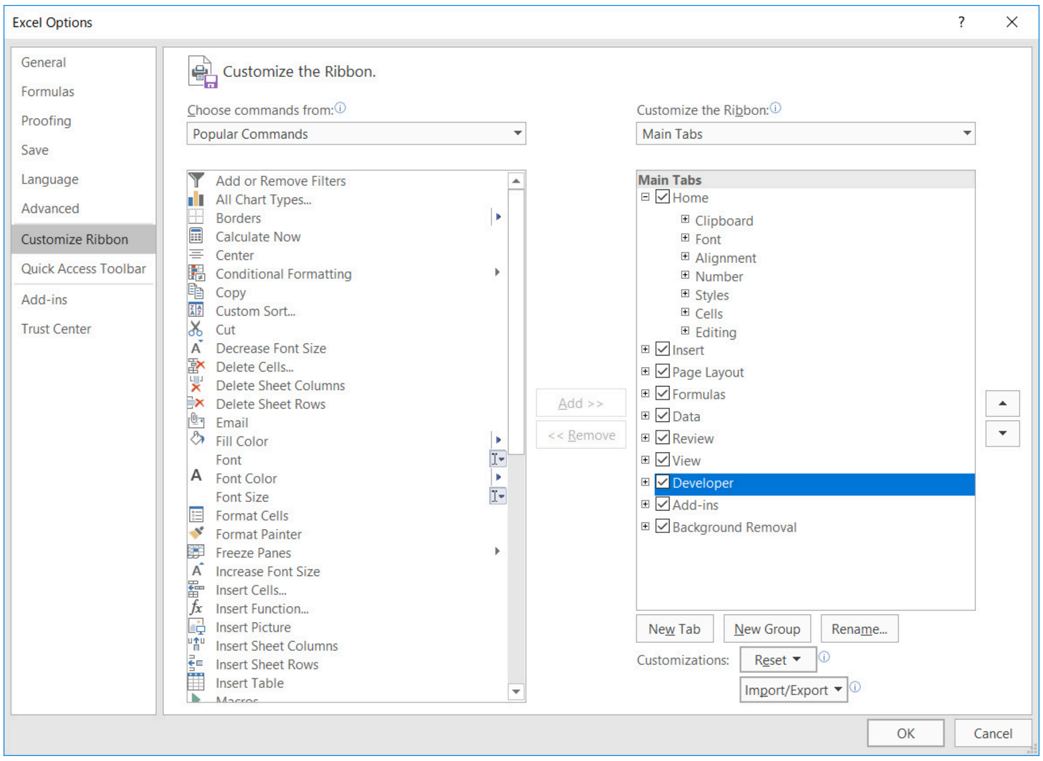
4 Troubleshooting

4.1 Developer ribbon does not appear

In the default, Microsoft Office Excel does not show the **Developer** ribbon. Complete the following procedure (in the case of Microsoft Office Excel 2016).

1. Click on the **File** ribbon.

2. Click on the **Option** ribbon.
3. Click on the **Customize Ribbons** in **Excel Options** window.
4. Confirm whether **Main Tab** appears in the **Customize Ribbons** field (Fig.5), and then look for **Developer** in the list of ribbons in **Main Tab**. Check the check box for **Developer**
5. Click on the **OK** button.
6. Close the **Excel's Option** window, and then the **Developer** button appears. If the button does not, then open the Excel file again.


Choose commands from: Popular Commands

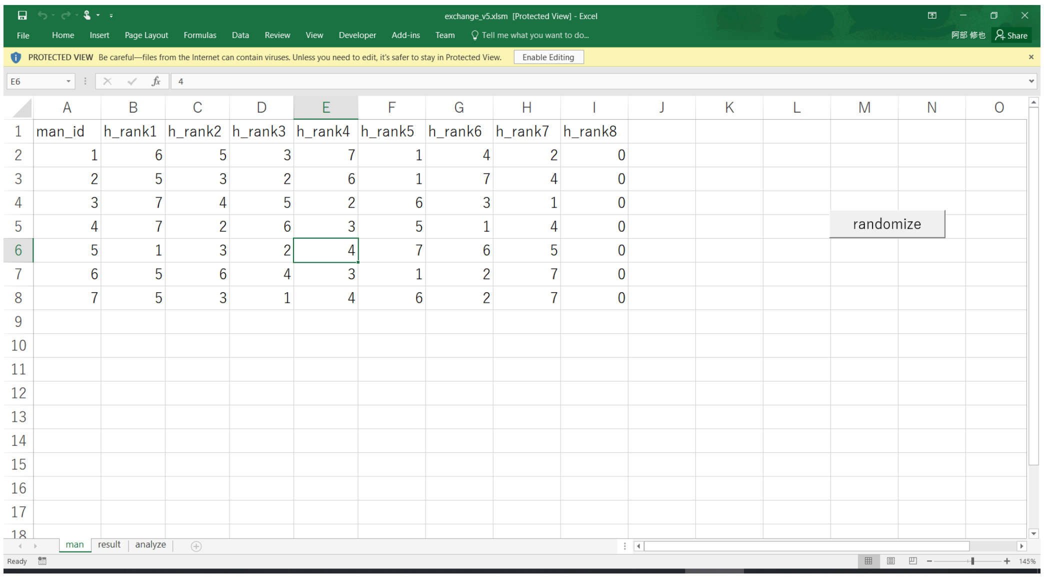
Customize the Ribbon: Main Tabs

Figure 5: A Screen-shot of Excel Options

4.2 Error messages appear

Confirm the following points, when error messages appear after pressing the buttons to run the program.

- Did you input data in Arabic numerals with single-byte characters in the VBA window and man sheet?
- Does the number of house owners initially set in the VBA window coincide with the one which appears in man sheet?



PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View.

man_id	h_rank1	h_rank2	h_rank3	h_rank4	h_rank5	h_rank6	h_rank7	h_rank8
1	6	5	3	7	1	4	2	0
2	5	3	2	6	1	7	4	0
3	7	4	5	2	6	3	1	0
4	7	2	6	3	5	1	4	0
5	1	3	2	4	7	6	5	0
6	5	6	4	3	1	2	7	0
7	5	3	1	4	6	2	7	0

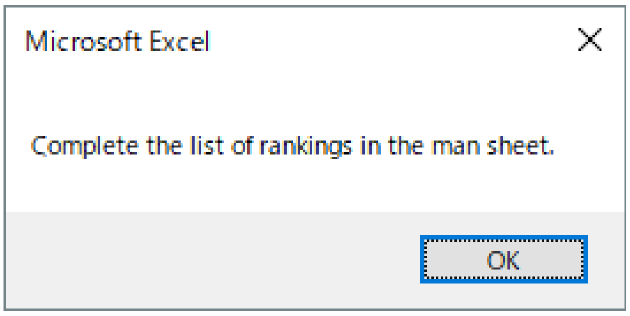
Figure 6: A Screen-shot of Protected View

4.3 No error message but incorrect allocation

Confirm the following points, when error messages appear but the software does not generate a correct allocation after pressing the buttons to run the program.

- Did you open your Excel file in Protected View (Fig.6)? In that case, click on the **Enable Editing** button to end Protected View.
- The previous result remains in the analyze sheet. Press the eval button in the result sheet for updating the result shown in the analyze sheet.
- If there is a blank space (missing value) in the man sheet, the following error message (Fig.7) appears, when you press the eval button in the result sheet.

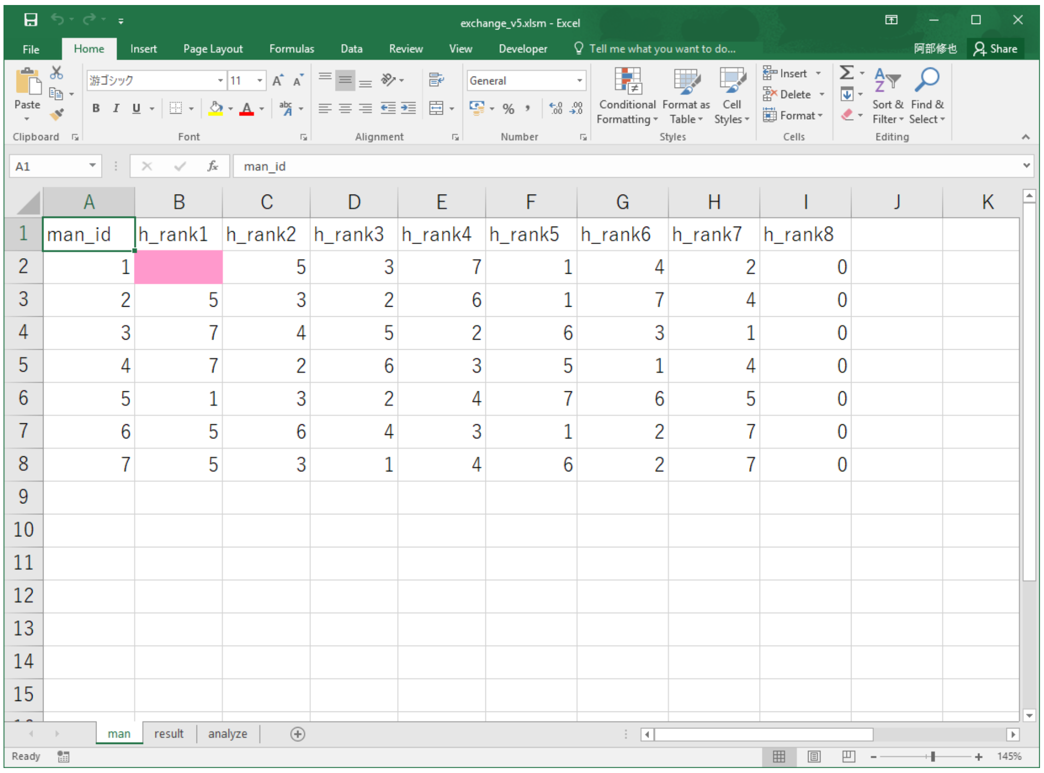
  Then, click on the OK button and open the man sheet. The blank space is indicated in pink (Fig.8) there .



Microsoft Excel

Complete the list of rankings in the man sheet.

Figure 7: An Error Message: A Blank Space in the Man Sheet



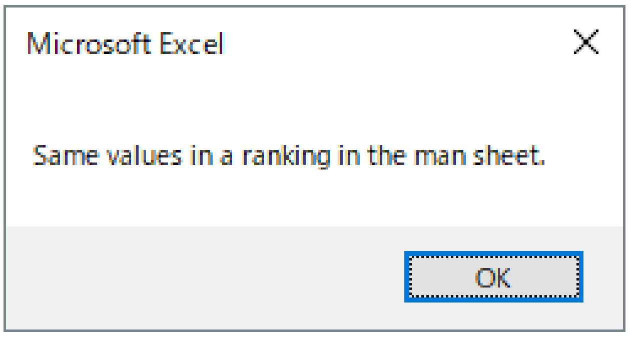
	A	B	C	D	E	F	G	H	I
1	man_id	h_rank1	h_rank2	h_rank3	h_rank4	h_rank5	h_rank6	h_rank7	h_rank8
2	1		5	3	7	1	4	2	0
3	2	5	3	2	6	1	7	4	0
4	3	7	4	5	2	6	3	1	0
5	4	7	2	6	3	5	1	4	0
6	5	1	3	2	4	7	6	5	0
7	6	5	6	4	3	1	2	7	0
8	7	5	3	1	4	6	2	7	0

Figure 8: A Blank Space Indicated in the Man Sheet

- Are there any multiple numbers for a man_id in the man sheet? In that case, the following error message (Fig.9) appears, when you press the eval button in the result sheet.

  Then, click on the OK button and open the man sheet. The multiple numbers are indicated in pink (Fig.10) there.

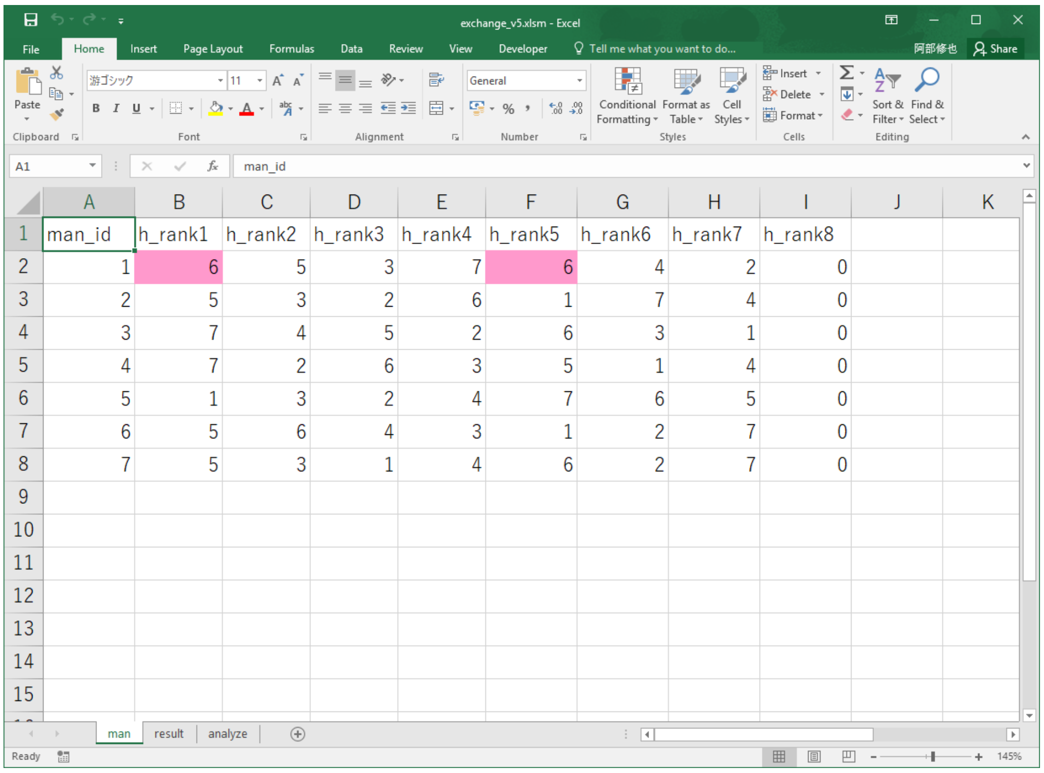
- You may leave a blank space (missing value) in the result sheet, when



Microsoft Excel

Same values in a ranking in the man sheet.

Figure 9: An Error Message: Multiple Values for a House Owner



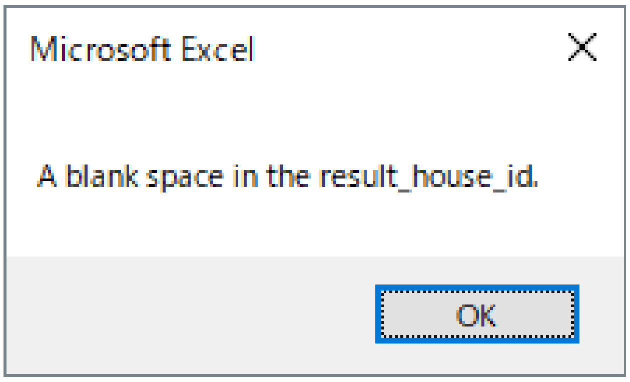
	A	B	C	D	E	F	G	H	I	J	K
1	man_id	h_rank1	h_rank2	h_rank3	h_rank4	h_rank5	h_rank6	h_rank7	h_rank8		
2	1	6	5	3	7	6	4	2	0		
3	2	5	3	2	6	1	7	4	0		
4	3	7	4	5	2	6	3	1	0		
5	4	7	2	6	3	5	1	4	0		
6	5	1	3	2	4	7	6	5	0		
7	6	5	6	4	3	1	2	7	0		
8	7	5	3	1	4	6	2	7	0		

Figure 10: Multiple Values Indicated in the Man Sheet

you enter some values by your own hand. In that case, the following error message (Fig.11) appears, when you press the eval button in the result sheet.

Then, click on the OK button and find the blank space indicated in pink (Fig.12) in the result sheet.

- You may enter the same number for several house owners in the result


Microsoft Excel

A blank space in the result_house_id.

Figure 11: An Error Message: A Blank Space in the Result Sheet

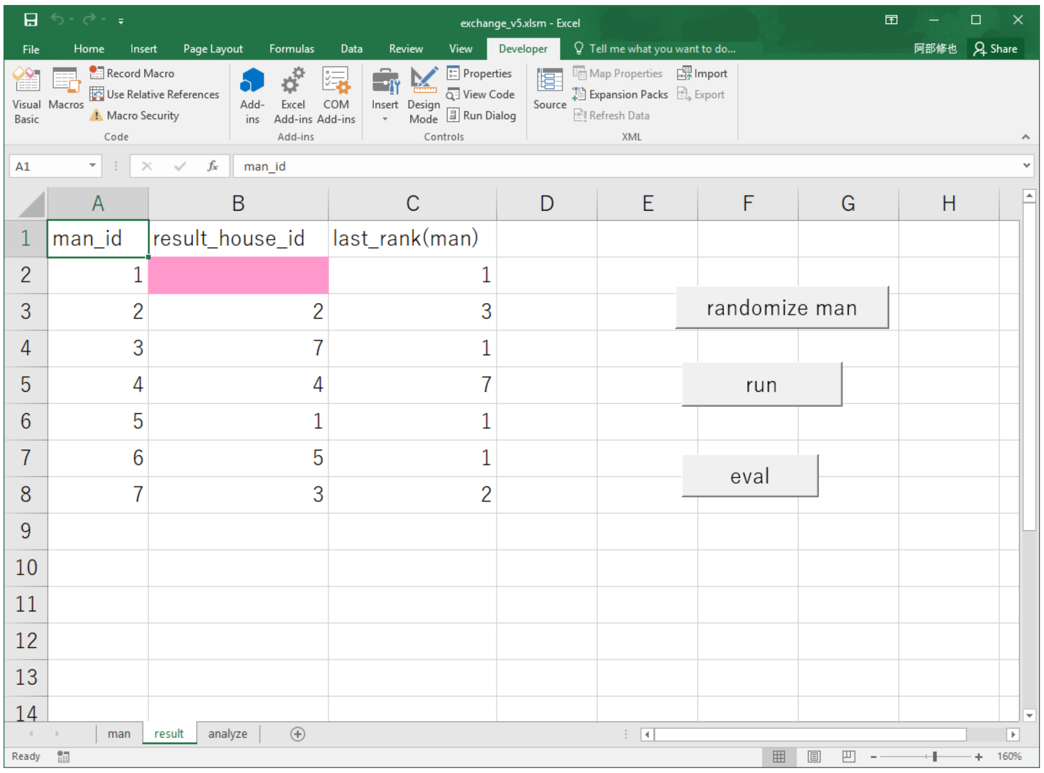
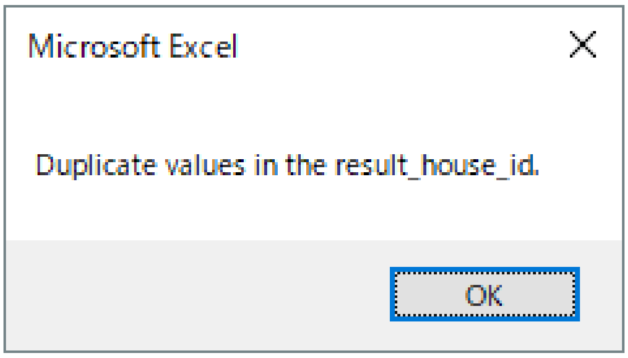

man_id	result_house_id	last_rank(man)
1	🟪	1
2	2	3
3	7	1
4	4	7
5	1	1
6	5	1
7	3	2

Figure 12: A Blank Space Indicated in the Result Sheet

sheet, when you enter the data by your own hand. In that case, the following error message (Fig.13) appears, when you press the eval button in the result sheet.

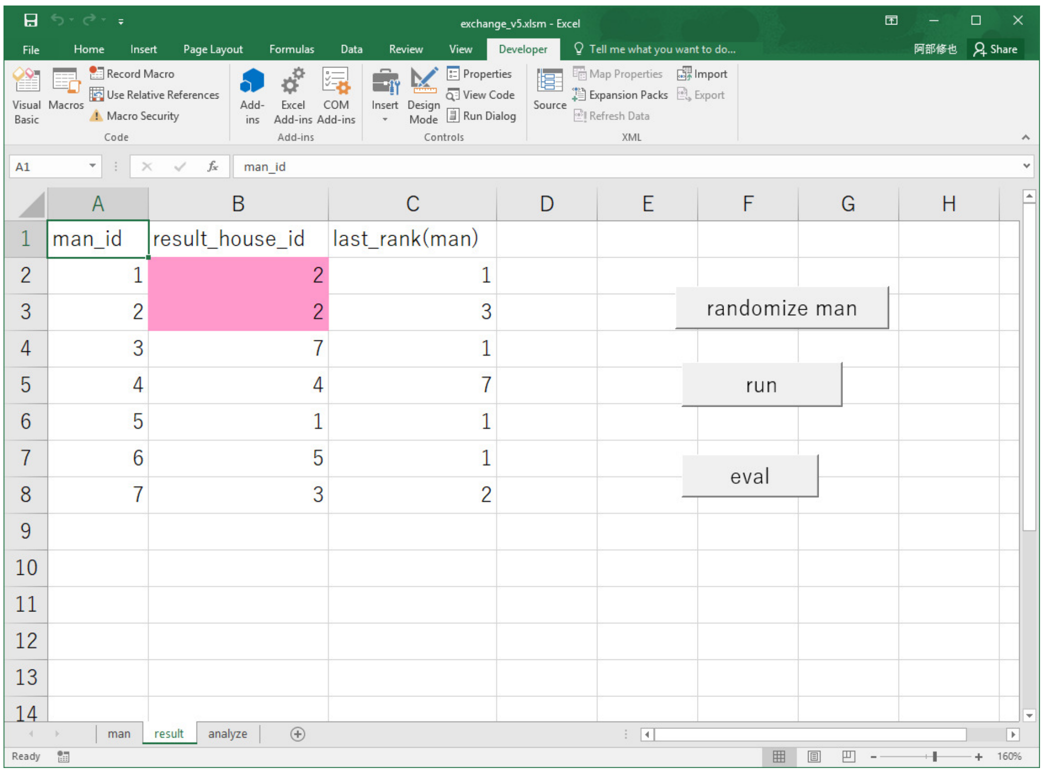
Then, click on the OK button and find the same numbers indicated in pink (Fig.14) in the result sheet.



Microsoft Excel

Duplicate values in the result_house_id.

Figure 13: An Error Message: The Same Numbers in the Result Sheet



exchange_v5.xlsm - Excel

man_id	result_house_id	last_rank(man)
1	2	1
2	2	3
3	7	1
4	4	7
5	1	1
6	5	1
7	3	2

Figure 14: The Same Numbers Indicated in the Result Sheet

Appendix: A House Swapping Problem

Let $N = \{A, B, \ldots\}$ be the set of traders. Initially, A owns house a, B owns house b, and so on. The traders can transfer ownership of their houses among themselves in any way they wish, except that no one is allowed to own more than one house at the end. The traders have only strict ordinal preferences over houses, i.e., each has a simple ranking of all the houses, without any ties. There are no side payments; indeed, as this is a "barter economy", questions of monetary value do not arise.

In Table 4, for example, the preferences of traders over houses are listed, where the upper bar on a lower-case letter of the Roman alphabet marks the trader's own house; beyond that point the ordering does not matter. We see that while B likes her own house best, the others have possibilities of "trading up" to something better. For example, E and F would each be better off by a rank if they exchanged houses. This is a house-swapping problem.

Table 4: Table of Preferences over all houses

A:	c	e	f	$\bar{a}$	b	d
B:	$\bar{b}$	a	c	e	f	d
C:	e	f	$\bar{c}$	a	d	b
D:	c	a	b	e	$\bar{d}$	f
E:	d	c	b	f	$\bar{e}$	a
F:	b	d	e	$\bar{f}$	a	c

In every house-swapping problem, there is always at least one reassignment of the houses such that no group of traders could have done better for all of its members by trading their own houses only among themselves. Note that there may be several reassignments that satisfy this property. There is, however, only one reassignment of the houses so that no group could have improved even one member's position without making some other member worse off. This unique reassignment is called **strict core allocation** in game theory.

The following algorithm which was developed by David Gale and introduced in Shapley and Scarf (1974) is based on the ideas of recursively using cycles of traders' top choices, and it produces the unique strict core allocation for every house-swapping problem. We here use the above six-trader example described in Table 4 in order to illustrate the Top Trading Cycle algorithm, or shortly, **TTC algorithm**.

(Step 1) Draw a directed graph, as below (Fig.15), with each trader represented by a vertex from which an edge points to the owner of his or her top-tanked house.

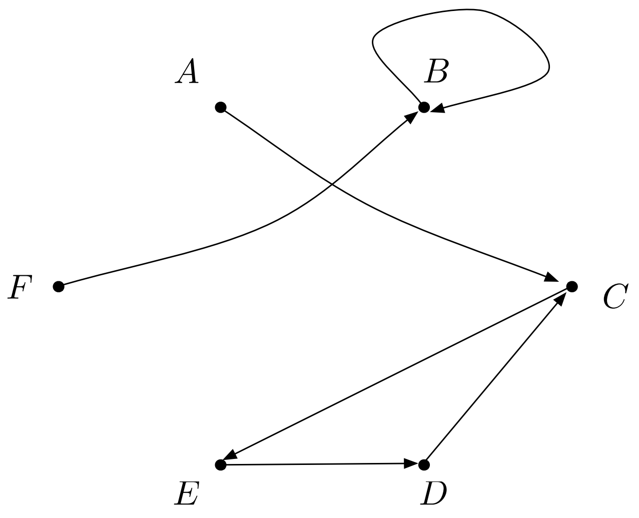


Figure 15: $T_1 = [CED]$, $T_2 = [B]$

(Step 2) Find the top trading cycle(s) by starting at any vertex and following the arrows until the path loops back on itself. In the example, starting at A, C, D, or E yields the circle $[CDE]$, and starting at B or F yields the circle $[B]$.

(Step 3) Delete from the preference table all the traders appearing in the TTCs discovered in Step 2, and return to Step 1 if any traders are left.

In the example, TTCs are $T_1 = [CED]$ and $T_2 = [B]$, and thus C, D, E, and B drop. Then, a new table of preferences is given as Table 5.

Table 5: Table of Preferences over two houses

A:	f	$\bar{a}$
F:	$\bar{f}$	a

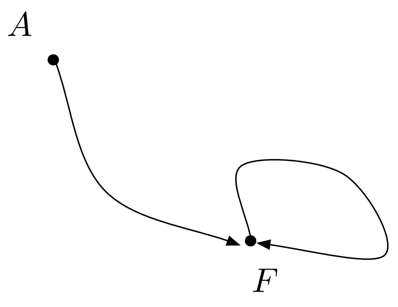


Figure 16: $T_3 = [F]$

Next, a TTC is $T_3 = [F]$ as depicted in Fig.16, and thus F drops. Then, a new table of preference is given as Table 6.

Table 6: Table of Preference over a house

A:	$\bar{a}$

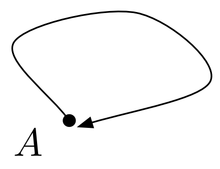


Figure 17: $T_4 = [A]$

(Step 4) Next, a TTC is $T_4 = [A]$ as depicted in Fig.17. When every trader has been assigned to a TTC in this manner, implement all the indicated trades. In other words, award to each trader the house originally owned by his or her successor in the TTC. Thus, in the example C gets e, E gets d, B gets b, etc. The final allocation is

$$\mathcal{A} =< Aa, Bb, Ce, Dc, Ed, Ff > .$$

The allocation $\mathcal{A}$ obtained by the TTC algorithm is stable in the sense that no coalition of traders, trading only with each other, could have achieved any allocation in which all members would be better off. Further, $\mathcal{A}$ is strongly stable in the sense that no coalition of traders, trading only with each other, can achieve an allocation in which at least one member is better off and none are worse off. Moreover, it is the only allocation of this kind.